Displaying and Interacting with Lists

Dr. Abdelkarim Erradi

CSE@QU

Outline

Displaying a List

Displaying a List

- In apps it is common to display collections of items

- For displaying a small collection of items, a Colum or Row layouts could be used

  - The **verticalScroll()** modifier could be used to make the <u>Column</u> scrollable

  - The **horizontalScroll()** modifier could be used to make the Row scrollable

- For displaying a large list, using a Column/Row layout can cause performance issues

  - since all the items will be composed and laid out whether or not they are visible

  - Use <u>LazyColumn</u> and <u>LazyRow</u> to only compose and lay out items which are **visible on screen**

Displaying a List

Making the [Column](#) scrollable by using the **verticalScroll()** modifier

```kotlin
@Composable
fun SurahsList(surahs: List<Surah>) {
    Column(modifier =
      Modifier.verticalScroll(rememberScrollState())
    ) {
        if (surahs.isEmpty()) {
            Text("Loading surahs failed.")
        } else {
            surahs.forEach {
                SurahCard(surah = it)
            }
        }
    }
}
```

Common Modifiers

- `Column(modifier = Modifier.verticalScroll(rememberScrollState())`
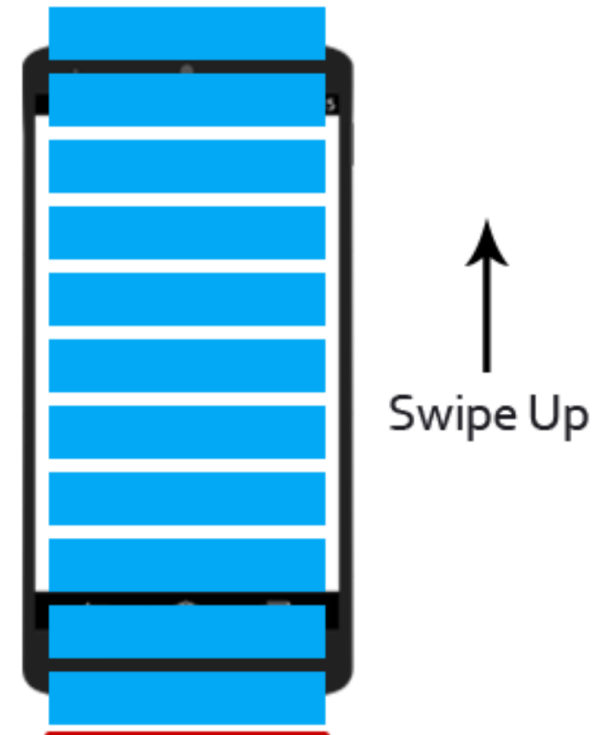
Makes the column scrollable

- `Row(modifier = Modifier.horizontalScroll(rememberScrollState())`

Makes the row scrollable

- `Modifier.fillMaxWidth() / .fillMaxHeight() / .fillMaxSize()` - occupy the available space

What is a LazyColumn?

- LazyColumn is scrollable container for displaying a list of composables

- A flexible container for efficiently displaying, and interacting with large sets of data

  - As user scrolls, views are created to display new items

  - Efficient as it uses a limited number of views

- **LazyColumn** produces a vertically scrolling list, and **LazyRow** produces a horizontally scrolling list.

Swipe Up

LazyColumn methods

- **LazyColumn** provides a number of functions for describing items in the layout:
  - `item()` to add a single item (e.g., header/footer)
  - `items(aList)` to add multiple items
  - `itemsIndexed(aList)` to add multiple items and provides an index

```
import androidx.compose.foundation.lazy.items
…
LazyColumn {
    items(surahs) {
        SurahCard(it)
    }
}
```

Styling a List - Content spacing

- Use **Arrangement.spacedBy()** to add spacing in-between items

```
LazyColumn(
    verticalArrangement = Arrangement.spacedBy(4.dp),
) {}
```

- Similarly for LazyRow:

```
LazyRow(
    horizontalArrangement = Arrangement.spacedBy(4.dp),
) {}
```

Styling a List – Content padding

- To add padding around the edges of the content to pass some **PaddingValues** to the **contentPadding** parameter

```
LazyColumn(
    contentPadding = PaddingValues(horizontal = 16.dp, vertical = 8.dp)
) {}
```

- E.g., add 16.dp of padding to the horizontal edges (left and right), and then 8.dp to the top and bottom of the content

- Padding is applied to the content, not to the LazyColumn itself

```kotlin
LazyColumn(contentPadding =
    PaddingValues(horizontal = 8.dp, vertical = 8.dp),
    verticalArrangement = Arrangement.spacedBy(8.dp)
) {
    item {
        Text(
            text = "سور القرآن الكريم",
            textAlign = TextAlign.Center,
            modifier = Modifier.fillMaxWidth(),
            style = TextStyle(
                fontWeight = FontWeight.Bold,
                fontSize = 24.sp,
                color = Color.Blue,
                textDirection = TextDirection.Rtl
            )
        )
    }
    items(surahs) {
        SurahCard(it)
    }
    item {
        Text(
            text = "$surahCount سورة - $ayaCount آية",
            textAlign = TextAlign.Center,
            modifier = Modifier.fillMaxWidth(),
            style = TextStyle(
                fontWeight = FontWeight.Bold,
                fontSize = 20.sp,
                color = Color.Blue,
                textDirection = TextDirection.Rtl
            )
        )
    }
}
```

Interacting with a List

Search

Sort

Handling Item Clicked Event

Add/Update a list item

Swipe to Delete and Undo

Pull to refresh

Sticky headers

Summary

- Use the **verticalScroll** or **horizontalScroll** modifiers to display a small list of composables in a Column or a Row

- For dynamic and larger lists use **LazyColumn** and **LazyRow** for the vertical and horizontal scenarios, respectively

- You can program various interactions with a displayed list including *search*, *sort*, *refresh*, *add*, *update* and *delete*

Resources

- Displaying a list using Jetpack Compose

https://developer.android.com/jetpack/compose/lists